Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of the claims in the application:

Listing of Claims:

1. (Currently Amended) A method including:

maintaining a state machine to provide a multi-bit output, each bit of the multi-bit output indicating a respective status of an associated thread of multiple threads being executed with a multithreaded processor;

detecting a change of status for a first thread within the multithreaded processor; and

responsive to the change of status for the first thread within the multithreaded processor, configuring a functional unit within the multithreaded processor in accordance with the multi-bit output of the state machine, wherein the configuring of the functional unit within the multithreaded processor includes inserting a fence instruction into an instruction stream for the first thread at a location proximate a front-end of the multithreaded processor, the fence instruction defining an event boundary within the instruction stream that assumes all memory accesses have drained from the multithreaded processor.

2. (Original) The method of claim 1 wherein each bit of the multi-bit output indicates the status of the associated thread as being active or inactive.

3. (Currently Amended) The method of claim 1 ~~2~~ wherein the configuring of the functional unit ~~comprises~~ includes partitioning the functional unit to service both the first thread and a second thread within the multithreaded processor when the change of status for the first thread comprises a transition from an inactive state to an active state.

4. (Currently Amended) The method of claim 1 ~~2~~ wherein the configuring of the functional unit ~~comprises~~ includes un-partitioning the functional unit to service a second thread, but not the first thread, within the multithreaded processor when the change of the status of the first thread comprises a transition from an active state to an inactive state.

5. (Original) The method of claim 1 wherein the detecting of the change in the status of the first thread comprises detecting the occurrence of an event for the first thread.

6. (Original) The method of claim 5 including asserting a first signal responsive to the occurrence of the event for the first thread, and evaluating the state machine during the assertion of the first signal.

7. (Original) The method of claim 6 wherein the functional unit within the multithreaded processor is configured, in accordance with the multi-bit output of the state machine, on the de-assertion of the first signal.

8. (Original) The method of claim 1 wherein the detecting of the change in the status of the first thread comprises detecting the occurrence of a sleep event for the first thread that transitions the first thread from an active state to a sleep state.

9. (Original) The method of claim 8 including, responsive to the detection of the occurrence of the sleep event, setting an inhibit register to inhibit an event that is not a

break event for the sleep state of the first thread.

10. (Currently Amended) The method of claim 1 wherein the configuring of the functional unit within the multithreaded processor ~~comprises~~ includes saving and deallocating state within the multithreaded processor for the first thread.

11. (Original) The method of claim 10 wherein the saving and deallocating of the state within the multithreaded processor for the first thread comprises recording the state for the first thread within a memory resource.

12. (Currently Amended) The method of claim 1 wherein the configuring of the functional unit within the multithreaded processor ~~comprises~~ includes making registers, within a register file of the multithreaded processor, available to a second thread within the multithreaded processor.

13. (Original) The method of claim 1 wherein the functional unit comprises any one of the group of functional units including a memory order buffer, a store buffer, a translation lookaside buffer, a reorder buffer, a register alias table, and a free list manager.

14. (Cancelled) The method of claim 1 wherein the configuring of the functional unit includes inserting a fence instruction into an instruction stream for the first thread at a location proximate a front-end of the multithreaded processor, the fence instruction defining an event boundary within the instruction stream that assumes all memory accesses have drained from the processor.

15. (Original) The method of claim 1 wherein the configuring of the functional unit includes restoring state within the multithreaded processor.

16. (Original) The method of claim 1 wherein the detecting of the change in the status of the first thread comprises detecting the occurrence of a break event for the first thread that transitions the first thread from a sleep state to an active state.

17. (Original) The method of claim 16 including detecting a third event for the first thread that does not constitute a break event, and logging the third event within a pending register associated with the first thread.

18. (Currently Amended) Apparatus ~~comprising~~ including:

a state machine to provide a multi-bit output, each bit of the multi-output indicating a respective status of an associated thread of multiple threads being executed within a multithreaded processor, and to detect a change of status for a first thread within the multithreaded processor; and

configuration logic to configure a functional unit within the multithreaded processor in accordance with the multi-bit output of the state machine; and

a microcode sequencer to introduce a fence instruction into an instruction stream for the first thread at a location proximate a front-end of the multithreaded processor, the fence instruction defining an event boundary within the instruction stream to ensure that all memory accesses drain from the multithreaded processor.

19. (Original) The apparatus of claim 18 wherein each bit of the multi-bit output indicates the status of the associated thread as being active or inactive.

20. (Currently Amended) The apparatus of claim 19 wherein the configuration logic ~~partitions~~ is to partition the functional unit to service both the first thread and a second thread within the multithreaded processor when the change of status for the first thread comprises a transition from an inactive state to an active state and the second thread is in an active state.

21. (Cancelled) The apparatus of claim 19 wherein the configuration logic un-partitions the functional unit to service a second thread, but not the first thread, within the multithreaded processor when the change of the status of the first thread comprises a transition from an active state to an inactive state and the second thread is in an active state.

22. (Currently Amended) The apparatus of claim 18 wherein the state machine ~~detects~~ is to detect the change in the status of the first thread by detecting the occurrence of an event for the first thread.

23. (Currently Amended) The apparatus of claim 22 including an event detector ~~that asserts~~ to assert a clearing signal responsive to the occurrence of the event for the first thread, and wherein the state machine is evaluated during the assertion of the first signal.

24. (Currently Amended) The apparatus of claim 23 wherein the configuration logic ~~configures~~ is to configure the functional unit within the multithreaded processor in accordance with the multi-bit output of the state machine on the de-assertion of the clearing signal.

25. (Original) The apparatus of claim 18 wherein the state machine, to detect the change in the status of the first thread, detects the occurrence of a sleep event for the first

thread that transitions the first thread from an active state to a sleep state.

26. (Original) The apparatus of claim 25 including a microcode sequencer that, responsive to the detection of the occurrence of the sleep event, issues a microinstruction to set an inhibit register to inhibit an event that is not a break event for the sleep state of the first thread.

27. (Currently Amended) The apparatus of claim 18 wherein the configuration logic ~~saves, deallocates and restores~~ is to save, deallocate and restore state within an associated functional unit for the first thread.

28. (Currently Amended) The apparatus of claim 27 wherein the configuration logic associated with the functional unit ~~records~~ is to record state information for the first thread within a memory resource to save and deallocate state, and restores state information for the first thread to functional unit from the memory resource to restore state.

29. (Original) The apparatus of claim 27 wherein the configuration logic associated with the functional unit makes registers, within a register file of the multithreaded processor, allocated to the first thread available to a second thread within the multithreaded processor if the first thread exits and makes registers, within the register file of the multithreaded processor, allocated to the second thread available to the first thread within the multithreaded processor if the second thread exits.

30. (Currently Amended) The apparatus of claim 18 wherein the functional unit ~~comprises~~ includes any one of the group of functional units including a memory order buffer, a store buffer, a translation lookaside buffer, a reorder buffer, a register alias table,

and a free list manager.

31. (Cancelled) The apparatus of claim 18 including a microcode sequencer that introduces a fence instruction into an instruction stream for the first thread at a location proximate a front-end of the multithreaded processor, the fence instruction defining an event boundary within the instruction stream to ensure that all memory accesses drain from the processor.

32. (Currently Amended) The apparatus of claim 18 wherein ~~the configuring of the functional unit includes restoring~~ the configuration logic is to restore state within the multithreaded processor.

33. (Currently Amended) The apparatus of claim 23 wherein the event detector ~~detects~~ is to detect the change in the status of the first thread by detecting the occurrence of a break event for the first thread that transitions the first thread from a sleep state to an active state.

34. (Currently Amended) The apparatus of claim 23 wherein the event detector ~~detects~~ is to detect a third event for the first thread that does not constitute a break event, and logs the third event within a pending register associated with the first thread.

35. (Currently Amended) Apparatus comprising:

first means for providing a multi-bit output, each bit of the multi-output indicating a respective status of an associated thread of multiple threads being executed within a multithreaded processor, and to detect a change of status for a first thread

within the multithreaded processor; and

second means for configuring a functional unit within the multithreaded processor in accordance with the multi-bit output of the state machine, wherein the configuring of the functional unit within the multithreaded processor includes inserting a fence instruction into an instruction stream for the first thread at a location proximate a front-end of the multithreaded processor, the fence instruction defining an event boundary within the instruction stream that assumes all memory accesses have drained from the multithreaded processor.

36. (Currently Amended) A machine-readable medium including a sequence of instructions that, when executed by a machine, cause the machine to:

maintain a state machine to provide a multi-bit output, each bit of the multi-bit output indicating a respective status of an associated thread of multiple threads being executed with a multithreaded processor;

detect a change of status for a first thread within the multithreaded processor; and

configure a functional unit within the multithreaded processor in accordance with the multi-bit output of the state machine, wherein the configuring of the functional unit within the multithreaded processor includes inserting a fence instruction into an instruction stream for the first thread at a location proximate a front-end of the multithreaded processor, the fence instruction defining an event boundary within the instruction stream that assumes all memory accesses have drained from the multithreaded processor.